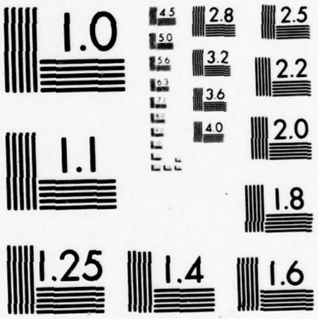
AD-A071 945 COLORADO STATE UNIV FORT COLLINS DEPT OF MATHEMATICS F/G 9/2
DATA FITTING: SOME ADAPTIVE METHODS.(U)
FEB 79 G D TAYLOR AFOSR-76-2878
UNCLASSIFIED AFOSR-TR-79-0858 NL

| OF |
AD
A071945

END
DATE
FILMED
9-79
DDC

END
DATE
FILMED
9-79
DDC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

Level II

DATA FITTING: SOME ADAPTIVE METHODS

G. D. Taylor

1. Introduction

Considerable effort has been, and is continuing to be spent developing numerical data fitting methods. In view of our increasing ability to collect immense and diverse sets of data, the fact that data obtained from an experiment or process is usually not in a convenient form for immediate interpretation and that frequently the data is noisy due to instrumental or human error, continued research on this topic is essential. This continued research should lead to both a deeper understanding of the problems associated with numerical data fitting and to the development of numerical data fitting software packages. The complexity of this area of research is directly related to the complexity and variety of data sets that can be encountered.

Thus, there is a need for numerical data fitting software packages which allow for effective data reduction of massive data sets for computational manipulation, for interpolation, for the inclusion of specific constraints (e.g. monotoneness) that the mathematical representation must satisfy, etc. To date there exists no single software package capable of processing any given data set in even the one-dimensional setting. In fact, the development of such an all-purpose black box is probably not feasible today nor in the near future. In this regard we agree with the comment made by J. R. Rice [18] that it is irrational to expect one algorithm to excel in all respects as a general purpose curve fitting routine. At present, it should probably be required that all existing packages be prefaced with a caution that the particular package may fail to return an acceptable fit to certain data sets. For example, in the case of one-dimensional sparse data, a given package may fit the data to within any tolerance desired by the user but exhibit gross oscillatory behavior between the data points. One approach that might avoid many of these difficulties is to use an interactive data fitting package with graphics capabilities. With such capabilities, the preformance of the package on a given data set could be monitored during use or at least upon completion (possibly only when failure is suspected or observed). Better yet, such a

Research Sponsored by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant AFOSR-76-2878C and the National Science Foundation under Grant MCS-78-05847.

DATA FITTING: SOME ADAPTIVE METHODS

DATA FITTING: SOME ADAPTIVE METHODS

G. D. Taylor

1. Introduction

Considerable effort has been, and is continuing to be spent developing numerical data fitting methods. In view of our increasing ability to collect immense and diverse sets of data, the fact that data obtained from an experiment or process is usually not in a convenient form for immediate interpretation and that frequently the data is noisy due to instrumental or human error, continued research on this topic is essential. This continued research should lead to both a deeper understanding of the problems associated with numerical data fitting and to the development of numerical data fitting software packages. The complexity of this area of research is directly related to the complexity and variety of data sets that can be encountered.

Thus, there is a need for numerical data fitting software packages which allow for effective data reduction of massive data sets for computational manipulation, for interpolation, for taking into account specific conditions (e.g., nonnegativity) that the mathematical representation must satisfy, etc. To date there exists no single software package capable of processing any given data set in even the one-dimensional setting. In fact, the development of such an all purpose black box is probably not feasible today nor in the near future. In this regard we agree with the comment made by J. R. Rice [15] that it is irrational to expect one algorithm to excel in all respects as a general purpose data fitting routine. At present, it should probably be required that all existing packages be prefaced with a caution that the particular package may fail to return an acceptable fit to certain data sets (for example, in the case of one-dimensional spline data, a given package [illegible] any tolerance desired by [illegible] between the data points. One approach [illegible] difficulties is to use an interactive data fitting package with graphics capabilities. With such capabilities, the performance of the package on a given data set could be monitored during use or at least upon completion (possibly only when failure is suspected or observed). Better yet, such a

Research supported by the Air Force Office of Scientific Research, Air Force Systems Command, USAF, under Grant No. [illegible] and the National Science Foundation under Grant [illegible] package could allow for intelligent guidance during or prior to the numerical processing of the data.

In this paper, we wish to primarily discuss our efforts to develop adaptive numerical software for data fitting. Thus our aim here is not to attempt a survey of current numerical software for data fitting.

We shall divide our discussion into two sections. In the first section we shall discuss one-dimensional data fitting and in the second section we shall discuss multi-dimensional data fitting. For the one-dimensional setting, we have developed four adaptive numerical software packages [1, 10, 11]. We are now beginning a study focusing on numerical testing and comparing these packages and other packages currently available for fitting one-dimensional data. Here we would like to be able to classify or characterize properties of a given data set which makes a particular package superior or inferior. Features that lead to successful packages should possibly be incorporated into the more effective packages where feasible and development of interactive packages should be pursued for those packages that appear to be most effective. In addition, a survey of current numerical software curve fitting packages together with reporting actual experience in using the packages could be very useful.

For the multi-dimensional setting, efforts are currently underway to develop an adaptive numerical software package for fitting multi-dimensional data sets. This package is of a slightly different philosophy than the one-dimensional codes mentioned above and is based upon the least squares approximation operator. It is anticipated that this package will be available in the late spring of this year.

2. One-Dimensional Data Fitting

In the last few years, considerable interest has been directed towards developing one-dimensional data fitting techniques [1,3,4,5,6,7,10,11,12, 13,14,15,17,18,19]. This activity has resulted in the development of algorithms intended for interactive use [3,6,13] and, more recently, adaptive curve fitting packages that produce piecewise polynomial fits have appeared [1,10,11,12,17]. These latter software packages are adaptive in the sense that the knots or joining points of the individual polynomial pieces are automatically inserted by the code as needed.

The four adaptive numerical software curve fitting packages described in [1,10,11] all utilize the same adaptive strategy. These packages differ in the mode of approximation that is used to determine each piece of the piecewise polynomial fit to the data. The four modes of approximation are: (i) uniform approximation [10], (ii) least squares approximation [1,10], (iii) uniform restricted range approximation [11] and (iv) ℓ_1 approximation [1]. For each of these packages, the user provides the data, the degree of the polynomial pieces desired, the overall smoothness the final piecewise polynomial must exhibit on some interval containing the independent domain of the data and a tolerance which is used to determine when the desired accuracy of fit is achieved. In addition, for the uniform restricted range approximation code the user may specify a range of values (in the ordinate direction) about each data point through which the fit must pass. (i.e. At each point (x_i,y_i) the user may specify values $\ell_i \leq y_i \leq u_i$, $\ell_i < u_i$, for which the computed piecewise polynomial fit, $p(x)$, will satisfy $\ell_i \leq p(x_i) \leq u_i$.) If the user does not specify these values then they will be set automatically by the code using a strategy that roughly allows these values to differ in a manner proportional to the variation of the data.

In what follows let us assume that we wish to approximate the data $\{(x_i, y_i)\}_{i=1}^{M}$. Set $X = \{x_i\}_{i=1}^{M}$, $f(x_i) = y_i$, $i = 1, \ldots, M$, $a=\min\{x:x \in X\}$ and $b = \max\{x: x \in X\}$. Further, assume that one wishes to approximate this data with an error tolerance of at most ε by piecewise polynomials of degree $n - 1$ and overall smoothness on $[a, b]$ of s, $s < n - 1$. In this setting, each of these codes will calculate an approximation, p, to the data and a set of knots $\{t_i\}_{i=1}^{k} \subset X$ with $a = t_0 < t_1 < \ldots < t_k = b$ such that p restricted to $[t_{i-1}, t_i]$ is a polynomial $p_i \in \Pi_{n-1} = \{q$: q is a real algebraic polynomial of degree $\leq n - 1\}$, p has s continuous derivatives on $[a, b]$ and $\max\{|f(x) - p(x)|: x \in X\} \leq \varepsilon$. (For the restricted range code we will also have that $\ell_i \leq p(x_i) \leq u_i$ for $i = 1, \ldots, M$.)

Each code begins by calculating t_1. The calculation of the knots, t_i, is the adaptive feature of these codes and this is done in a left-to-right manner using a bisection strategy. Basically, the code first calculates $\tilde{t}_1$ to be essentially the largest point in X such that the best approximation (in the mode of the particular code being used), p_1, to f on $S_1 = [a, t_1] \cap X$ satisfies $\max\{|f(x) - p_1(x)|: x \in S_1\} \leq \varepsilon$. This is done by first setting $\tilde{a} = a$ and $c = \tilde{b} = b$ and finding the best approximation,

p_1, to f on $[a, \tilde{c}] \cap X$. If $\max\{|f(x) - p_1(x)|: x \in [a, \tilde{c}] \cap X\} \le \varepsilon$ then p_1 is the desired approximation and it is returned and the algorithm terminates. If this is not the case, then a new $\tilde{c} \in X$ is chosen where $\tilde{c}$ is a closest point in X to $\frac{1}{2}(\tilde{a} + \tilde{b})$, taken to be the larger point when there are two closest points in X to this value. Next, the best approximation to f on $[a, \tilde{c}] \cap X$ is calculated and the maximum absolute pointwise error on $[a, \tilde{c}] \cap X$ is found. If this error is $\le \varepsilon$ then $\tilde{a}$ is now set to be $\tilde{c}$, whereas in the contrary case $\tilde{b}$ is then set to be $\tilde{c}$. In this manner the set $[a, \tilde{a}] \cap X$ is always the largest current set on which we can approximate the data with a pointwise error $\le \varepsilon$ and $[a, \tilde{b}] \cap X$ is the smallest current set on which we cannot approximate the data with a pointwise error $\le \varepsilon$. This procedure is continued by again setting $\tilde{c} = \frac{1}{2}(\tilde{a} + \tilde{b})$ and continues until either $\tilde{a}$ and $\tilde{b}$ are adjacent points of X on $(\tilde{b} - \tilde{a}) \le \eta$, where η is a preset constant in the code. Finally, the algorithm set $\tilde{t}_1 = \tilde{a}$ when the above phase terminates. If s = 0 so that only continuity is required on [a, b], then t_1 is taken to be $\tilde{t}_1$. If s > 0, the algorithm then may "back off" from t_1 in an attempt to avoid the introduction of unwanted steep oscillations in the approximation. This "backing-off" procedure is probably very similar to that briefly mentioned by Rice [19] for his 1-pass algorithm. This procedure consists of examining the error function at the m = n - s - 1 largest extreme points, $\xi_1, \ldots, \xi_m$ of $(a, \tilde{t}_1] \cap X$ where ξ_ν being an extreme point means that $|f(\xi_\nu) - p_1(\xi_\nu)| \ge \mathrm{sign}(f(\xi_\nu) - p_1(\xi_\nu)) \cdot (f(x) - p_1(x))$ for $x = \max\{t \in X: t < \xi_\nu\}$ and $x = \min\{t \in X: t > \xi_\nu\}$. Then, t_1 is chosen to be the largest ξ_ν such that $|\tilde{f}'(\xi_\nu) - p_1'(\xi_\nu)|$ is less than a (user definable) prescribed tolerance, where $\tilde{f}'$ is the derivative of the quadratic interpolation of f centered at ξ_ν. If there does not exist such a ξ_ν, then t_1 is chosen to be the largest ξ_ν at which $|\tilde{f}'(\xi_\nu) - p_1'(\xi_\nu)|$ is a minimum. Our numerical experience indicates that this procedure contributes significantly towards the stability of the algorithm. The original motivation for this procedure was that in the theoretical case of approximating a continuously differentiable f on an interval we are guaranteed that f' and p_1' agree at interior extreme points. Thus, when we smoothly join the next polynomial piece, p_2, of our piecewise polynomial p to p_1 at t_1, this next piece will closely follow the direction of f at least near t_1.

Once the first knot t_1 has been chosen, the algorithm then repeats the above steps on the set $[t_1, b] \cap X$ provided $t_1 < b$ with the additional requirement that each best approximation, p_2, must now also satisfy the interpolatory constraints $p_2^{(j)}(t_1) = p_1^{(j)}(t_1)$, $j = 0, 1, \ldots, s$. The inclusion of these interpolatory constraints into the approximation operator is straightforward since they are imposed at the left-hand endpoint of the interval of approximation. Finally, special care is also taken in selecting the last knot t_{k-1} so that the set $[t_{k-1}, b] \cap X$ has sufficient points to allow for a reasonable approximation to be calculated on it.

One other common feature of these codes is the inclusion of a subroutine that will add artificial data points to a sparse data set. As mentioned earlier, approximation on a small (sparse) data set can lead to unacceptable approximations. Basically, what happens is that the approximant fits the sparse data very well but oscillates badly between the data points. In these cases we have found that the addition of artificial data points will sometimes remove this undesirable behavior. This optional subroutine will insert a continuous piecewise linear function through the data and then discretize this function to increase the size of the data set. Note that it is the user's responsibility to decide if piecewise linearization is really the best way to add artificial data points in his particular situation.

Based upon our general intuition of these various approximation operators, we suspect that the uniform adaptive curve fitting package should be used only on either precise data (e.g. from mathematical formulas) or on essentially noise-free data. The least squares adaptive curve fitting package should be effective for fitting data sets containing essentially normally distributed random noise. The uniform restricted range adaptive curve fitting package should allow the user the greatest control over the shape of the resulting fit (at a cost of having to understand the code to a greater depth). Here, the user can actually create restraining bands within which the resulting approximation will lie. Finally, the ℓ_1 adaptive curve fitting package should be effective for fitting large data sets that contain points which are very inaccurate with respect to the overall accuracy of the data. Finally, for sets suspected of having an occasion highly inaccurate data point, we have used a slightly different error tolerance check. This second error tolerance check says that the error tolerance criterion is not met when the pointwise absolute error is greater than ϵ at two consecutive points of X.

In the way of examples we have included seven figures illustrating the results of running algorithms (ii)-(iv). Figures 1 through 5 represent the results of running these algorithms on some oil shale data. Since relatively few data points (denoted by "x" in figures) were available, we filled in the gaps using our linear interpolation routine with 200-300 points. Figures 2 and 3 are the same approximation, the first one also plotting the restraining curves. Figures 6 and 7 are examples of approximating the exponential function with noise on 101 equally spaced points in [0, 2] where the second error tolerance check is used. All of these runs were done on Colorado State University's CDC CYBER 172. The run time was roughly the same for the ℓ_2 approximation code and the restricted range approximation code (taking roughly one-half second per subinterval needed) and the run time of the ℓ_1 approximation code was roughly two to three times greater.

We also approximated $\sqrt{x}$ on 201 equally spaced points in [0, 2] with algorithm (i) using $n = 6$, $s = 2$ and $\varepsilon = .01$. A piecewise polynomial fit was computed in 1.97 seconds, having an absolute error of .0093 at these points, with a total of 5 polynomial pieces. The knot locations were 0, .08, .230, .350, .810, 2.0. We chose this example to check the ability of the code to recover from a point where f(x) is difficult to approximate (here at $x = 0$). In way of comparison with straight least squares cubic spline (fixed knots) curve fitting we also approximated $\sqrt{x}$ on 201 equally spaced points using as our approximating family the set of all cubic splines with knots at $\{\frac{i}{10}\}_{i=0}^{20}$ and the routine ICSFKU in the IMSL library. The maximum error of this fit over the 201 points in [0, 2] used was .032 (at $x = 0$, the error improved towards 2) and the run time was 4.29 seconds. Using this same routine with only eleven equally spaced knots gave a maximum error of .0597 in a run time of 1.68 seconds. Finally, we also attempted to use the variable knot cubic spline fitting routine ICSVKU of the IMSL library with eleven knots on this problem. However, it failed to return satisfactory results (maximum error 14.79 in a run time of 13.2 seconds). In defense of this routine it should be noted that its documentation specifies that it is not intended for use in approximating precise mathematical functions and we initialized it with equally spaced knots.

In closing this section we wish to briefly mention three other adaptive curve fitting studies. The first is due to J. A. Payne [15]. In this paper an automatic curve fitting package is described. Basically the knots and the degrees of the polynomial pieces of the piecewise polynomial fit are selected in a left-to-right manner via a statistical criterion. Then the final fit is found by doing a simultaneously least squares fit of all the individual pieces on the whole data set.

In [17], J. R. Rice has developed an adaptive numerical software package based upon local Hermite interpolation rather than approximation on each subinterval to obtain each polynomial piece. This algorithm is primarily intended for approximating mathematical functions where one has access to a certain number of derivatives at each point. The adaptive strategy is similar to ours but without any backing-off procedure. In addition one can measure the error of approximation in any preselected L_p norm ($p \geq 1$). This code will return a smooth (user specified) piecewise polynomial and it will have fast run times.

In [12], an adaptive numerical software curve fitting routine that computes C^1 piecewise cubic polynomials is given. This routine is similar to our least squares routine with the major difference being in the "backing-off" procedure for selecting knots. They use a simpler procedure based upon ideas of Powell. Also, it should be noted that their algorithm is also applicable to on-line systems since a tolerance need not be prescribed in advance.

3. Multi-Dimensional Data Fitting

Considerable effort has been (and is being) expended developing methods of approximation of multi-dimensional data. A recent excellent survey by L. L. Schumaker [21] should be consulted for an overall view of the current state of the art in this topic. At present, according to J. R. Rice [20] in a talk given at the October 1978 SIAM meetings, there are no adaptive numerical software packages available for approximating multi-dimensional data sets. In this section, we will briefly discuss an adaptive multi-dimensional data fitting code currently being developed by C. R. Vogel and myself [22]. We hope to make this code available in the late spring. Our philosophy has been to develop a reasonably general code sometimes at the cost of efficiency. Thus, for example, the user will have a reasonable choice of possible basis functions and to increase this selection the user would have to modify the code itself. (We will probably include some directions for doing this). Also, the code will probably be set up for two-dimensional data with instructions on how to modify it to treat larger dimensional data sets. In addition, it will be based upon the least squares approximation operator only. For simplicity, we shall discuss the two-dimensional case in what follows.

Thus, given a two-dimensional scattered data set contained in a rectangle R, the code will attempt to calculate a piecewise polynomial of user prescribed degree and smoothness that fits the data with a least squares error less than or equal to a user prescribed tolerance. The respective polynomial pieces will be defined over subrectangles with sides parallel to the sides of R with each piece being of the form $\sum_{i=0}^{m} \sum_{j=0}^{m} a_{ij}x^iy^j$. To describe our adaptive strategy, assume that the algorithm has subdivided R into k subrectangles, $\{R_i\}_{i=1}^{k}$, such that $\bigcup_{i=1}^{k} R_i = R$ and $R_i \cap R_j$ is at most an edge for $i \neq j$. Then on this subdivision one solves a least squares approximation problem with equality constraints to find simultaneously $p_1, \ldots, p_k$ with p_i defined on R_i, that gives the best least squares approximation (of minimal least squares coefficient sum in the event of nonuniqueness) to the given data where the equality constraints correspond to the smoothness required. Setting p equal to the resulting piecewise polynomial approximation, the least squares norm of the difference of the data minus the value of p at the data point is calculated over each subrectangle. Each of these values is then compared to the user prescribed tolerance divided by $\sqrt{k}$. If for a particular subrectangle this value is greater than the prescribed tolerance divided by $\sqrt{k}$, then this rectangle must be subdivided on the next pass of the algorithm (provided the total number of subrectangles does not become too large). This rectangle is then divided into two subrectangles such that the (modulo a scaling factor) two subrectangles are as close to squares as possible. This, then forms a new collection of subrectangles on which a new least squares fit is computed as above. In the computation of these least squares fits we are using a weighted least squares approach based on Given's rotations as described in [8] rather than the Lagrange multiplier approach of [3]. At present we have a running FORTRAN code that corresponds to the above brief description. Currently, we are running numerical tests on this code and as mentioned earlier, we hope to make this code available for general use in the near future.

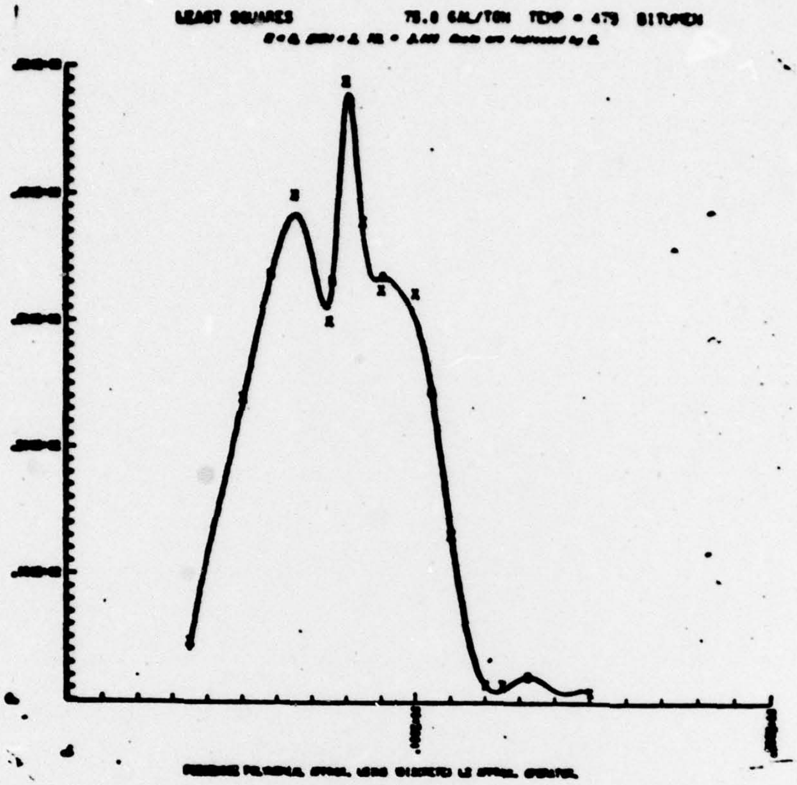


Figure 1

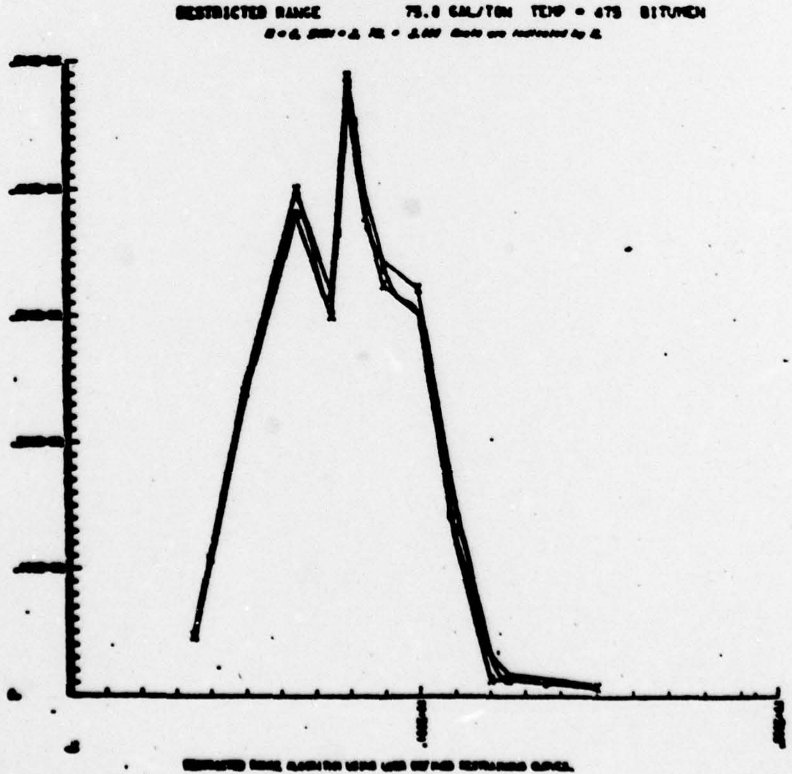


Figure 2

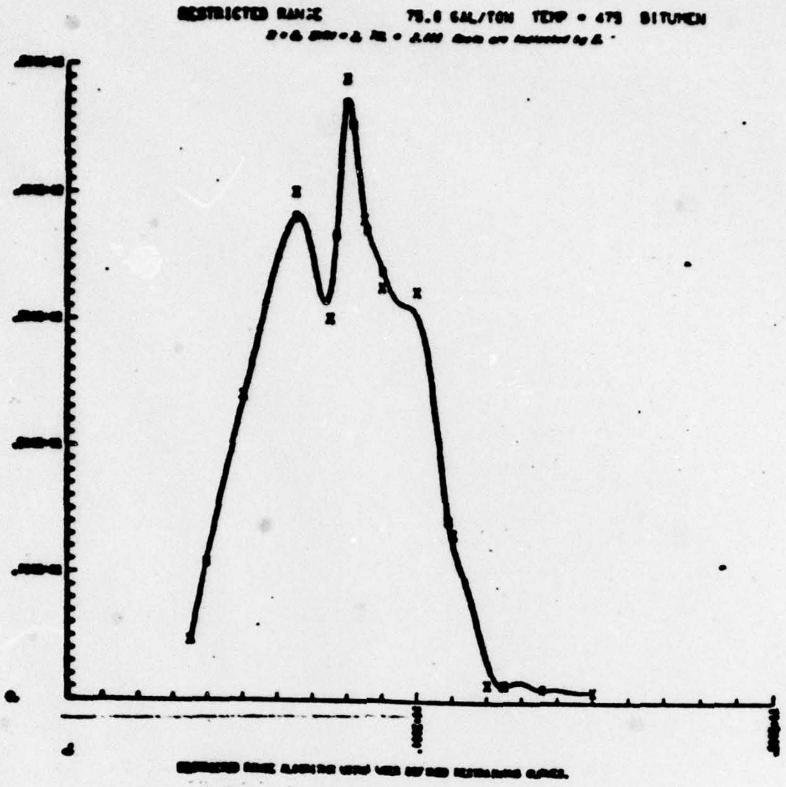


Figure 3

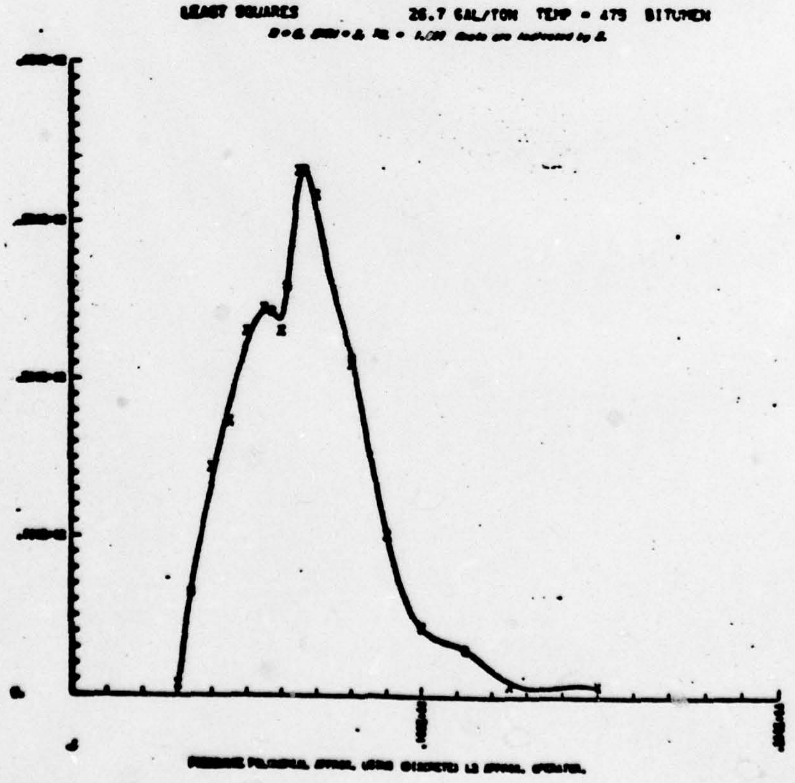


Figure 4

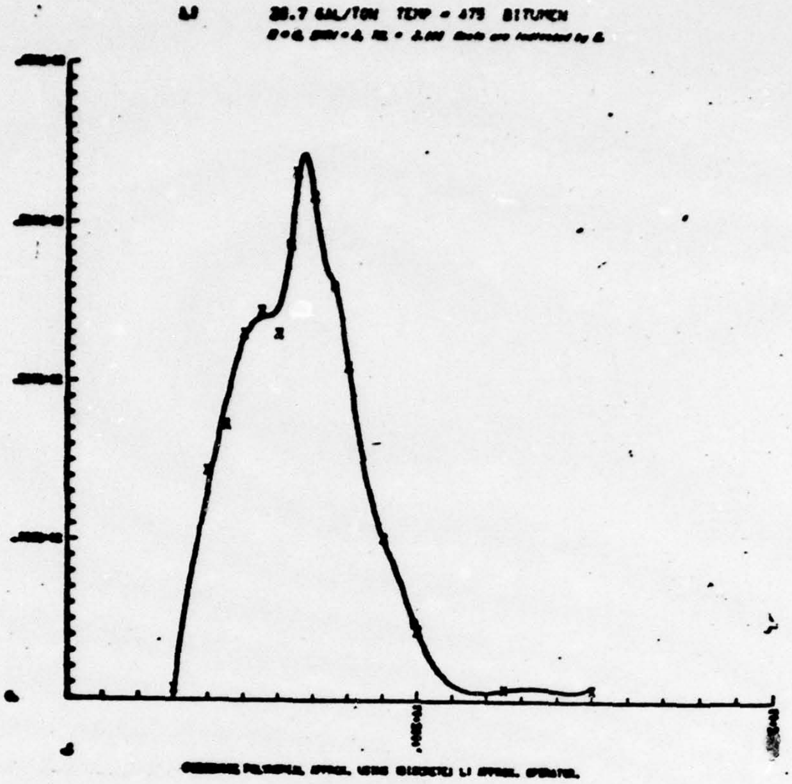


Figure 5

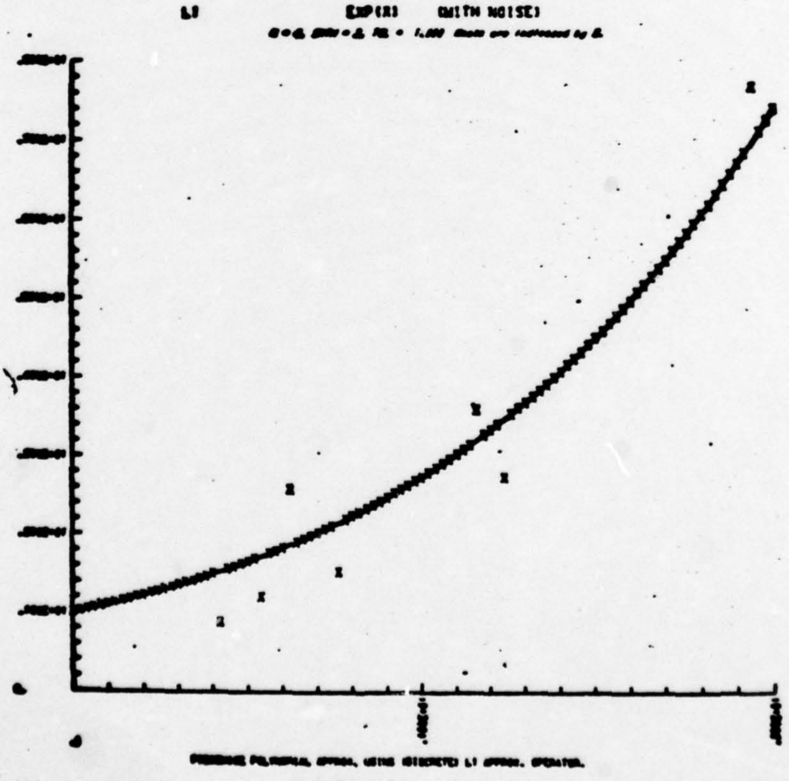


Figure 6

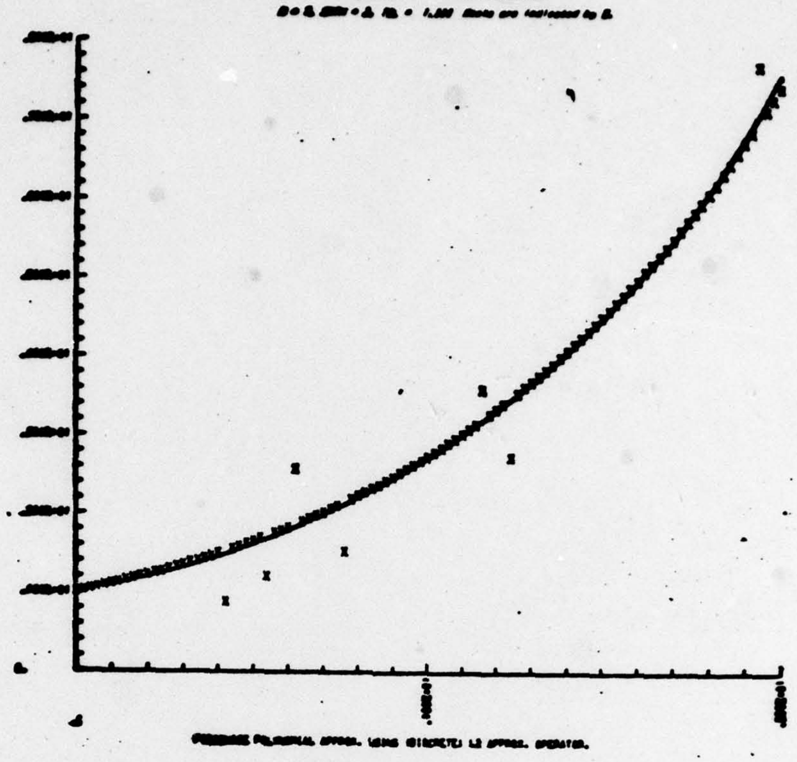

L2
EXP(X) (WITH NOISE)
[illegible] POLYNOMIAL APPROX. USING (DISCRETE) L2 APPROX. OPERATOR.

Figure 7

REFERENCES

1. P. G. Avila and G. D. Taylor, Adaptive ℓ_1-ℓ_2 curve fitting, to appear Internat. J. Numer. Methods Engrg.

2. R. H. Bartels, G. H. Golub and M. A. Saunders, Numerical techniques in mathematical programming, in Nonlinear Programming, Academic Press, Inc., 1970, pp. 123-175.

3. W. L. Chung, Automatic curve fitting for interactive display, Ph.D. Thesis, Department of Computer Science, University of Illinois, Urbana-Champaign, UIUCDCS-R-75-713, May 1975.

4. C. W. deBoor and J. R. Rice, Least square cubic spline approximation II - variable knots, CDS-TR21 Computer Science, Purdue University, 1968. Also IMSL subroutine ICSVKU.

5. L. E. Deimel, D. F. McAllister and J. A. Roulier, Smooth curve fitting with shape preservation using osculatory quadratic splines, preprint.

6. L. E. Deimel, C. L. Ross, R. J. Fornaro, D. F. McAllister and J. A. Roulier, Application of shape-preserving spline interpolation to interactive editing of photogrammetric data, preprint.

7. T. M. R. Ellis and D. H. McLain, Algorithm 514 - A new method of cubic curve fitting using local data, TOMS, 3 (1977), 175-178.

8. R. Hanson and C. Lawson, Solving Least Squares Problems, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1974.

9. J. G. Hayes and J. Halliday, The least squares fitting of cubic spline surfaces to general data sets, J. Inst. Maths. Applications 14 (1974), 89-103.

10. J. A. Hull and G. D. Taylor, Restricted range adaptive curve fitting, to appear Internat. J. Numer. Methods Engrg.

11. J. A. Hull and G. D. Taylor, Adaptive curve fitting, submitted.

12. K. Ichida, T. Kiyoni and F. Yoshimoto, Curve fitting by a one-pass method with a piecewise cubic polynomial, TOMS, 3 (1977), 164-174.

13. P. La Fata and J. B. Rosen, An interactive display for approximating by linear programming, Comm. ACM 13 (1970), 651-659.

14. D. F. McAllister and J. A. Roulier, Interpolation by convex quadratic splines, preprint.

15. J. A. Payne, An automatic curve-fitting package, Numerical Approximation to Functions and Data, J. G. Hayes editor, The Athlone Press, University of London, 1970, pp. 98-106.

16. M. D. J. Powell, Numerical methods for fitting functions of two variables, preprint, Computer Science and Systems Revision, A.E.R.E., Harwell, Oxfordshire, England.

17. J. R. Rice, ADAPT, Adaptive smooth curve fitting, Algorithm 525, TOMS, 4 (1978), 82-94.

18. J. R. Rice, General purpose curve fitting, Approximation Theory, A. Talbot editor, Academic Press, London, 1970, pp. 191-204.

19. J. R. Rice, Running orthogonalization, J. of Approximation Theory, 4 (1971), 332-338.

20. J. R. Rice, Adaptive multivariate approximation theory and applications, CSD-TR293, Computer Science, Purdue University, 1978.

21. L. L. Schumaker, Fitting surfaces to scattered data, Approximation Theory II, G.G.Lorentz editor, Acad.Press, New York, 1976,pp.203-268.

REFERENCES

22. G. D. Taylor and C. R. Vogel, An adaptive multi-dimensional data fitting package, in preparation.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. ~~REPORT NUMBER~~ 18 AFOSR-TR-79-Ø858	2. GOVT ACCESSION NO.
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 DATA FITTING: SOME ADAPTIVE METHODS •	5. ~~TYPE OF REPORT & PERIOD COVERED~~ 9 Interim rept.
	6. ~~PERFORMING ORG. REPORT NUMBER~~
7. AUTHOR(s) 10 G. D. Taylor	8. ~~CONTRACT OR GRANT NUMBER(s)~~ 15 ✓ AFOSR-76-2878, ✓NSF-MCS78-Ø5847
9. PERFORMING ORGANIZATION NAME AND ADDRESS Colorado State University Department of Mathematics Fort Collins, CO 80523	10. ~~PROGRAM ELEMENT, PROJECT~~, TASK AREA & WORK UNIT NUMBERS 16 17 61102F 2304 A3
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Office of Scientific Research/NM Bolling, AFB, D. C. 20332	12. ~~REPORT DATE~~ 11 Febr~~uary 1~~979
	13. NUMBER OF PAGES 14
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) 12 15p.	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Data fitting, adaptive data fitting.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper surveys some recent work of the author in developing numerical software for adaptive data fitting. Both one- and multi-dimensional data sets are considered.